

LA-UR-19-30822

Approved for public release; distribution is unlimited.

Title: Report on the Tin-II Thermal Neutron Detector

Author(s): Wender, Stephen Arthur
Couture, Aaron Joseph
Fairbanks, Thomas D.

Intended for: Report

Issued: 2019-10-24

Disclaimer:

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by Triad National Security, LLC for the National Nuclear Security Administration of U.S. Department of Energy under contract 89233218CNA000001. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

Report on the Tin-II Thermal Neutron Detector

Steve Wender, A. Couture, P-27, Tom Fairbanks, ISR-4

Los Alamos National Laboratory, Los Alamos NM 87545

Introduction

Thermal neutrons have recently become a concern in the semiconductor community where it has been reported that approximately 20% of the single-event upsets are due to thermal neutrons in some devices. The goal of this project is to measure the intensity of thermal neutrons in the High-Performance Computing (HPC) area. This is part of a larger effort to characterize the radiation environment in the HPC area which includes high-energy neutrons as well as thermal neutrons. To accomplish this goal, we designed and fabricated a neutron detector that is sensitive to thermal neutrons called Tin-II. The Tin-II detector consists of two identical ^3He ionization chambers with one detector bare and one detector shielded with cadmium. Because the cadmium shielding effectively blocks the thermal neutrons, the difference in count rates between these two detectors reflect the number of thermal neutrons detected. This report summarizes the status of our efforts to develop this detector.

We will discuss the design of the detector, the signal processing electronics, the data acquisition approach and initial commissioning of the detector. We will describe how we converted the number of counts/s to a measurement of the number of thermal neutrons/cm²/s.

The design of the detector is based on the Tinman detector that was designed to measure thermal neutrons in airplanes and was part of a Strategic Partnership Agreement with Honeywell, Inc.

Tin-II is now fully operational and has just been installed in the HPC area.

Background

Thermal neutrons are presently thought to be a threat to the reliable operation of semiconductor electronic devices. Thermal neutrons are produced when high-energy neutrons, which are produced by cosmic-ray interactions in the atmosphere, strike moderating material and lose energy to approximately 0.025 eV. These thermal neutrons can interact (capture reaction) with material in semiconductor devices and in some cases produce charged particles that can deposit charge in sensitive volumes of the device and produce upsets. Because the intensity of thermal neutrons depends on the particular environment, it is difficult to characterize the thermal neutron intensity without specifying the surrounding environment, so direct measurements are necessary. In particular, if we assume that high-energy cosmic-ray induced neutrons shower the HPC room, the intensity of thermal neutrons depends on the amount and proximity of moderating material. Therefore, we would expect more thermal neutrons near moderating material such as water and less thermal neutrons as we move farther away from moderating material.

A major concern is the boron content in semiconductor devices. Natural boron consists of two isotopes: ^{10}B (20% abundant) and ^{11}B (80% abundant). ^{10}B has a very large thermal-neutron cross section (3840

barns) and produces energetic charged particles (${}^7\text{Li}$ and α) that can deposit charge in semiconductor devices and therefore cause failures.

The effect of thermal neutrons on a semiconductor device depends on the product of the number of thermal neutrons present, “a flux”, and the effect of thermal neutrons on the device, “a cross section”. This effort is focused on determining the number of thermal neutrons in the HPC area. To determine the failure rate of particular devices due to thermal neutrons, it is necessary to determine the cross section for upsets due to thermal neutrons. The failure cross section can be determined in a subsequent measurement that can be performed at the LANSCE Lujan Center thermal-neutron source in the future.

Design of thermal neutron detector

We fabricated Tin-II to meet the requirements for installation in the HPC area. We based the design of Tin-II on the Tinman detector that was previously designed to fly in NASA aircraft to measure thermal neutrons in airplanes. Since the design of this detector was derived from the original Tinman detector, it shares many of the same design criteria. These design criteria included: low count rates, low power consumption, robust packaging and automated, unattended operation.

The detector was designed to operate at 28 volts DC because that was the available power on the aircraft. In Tin-II, the 28 volts is supplied by an external power supply. The detector draws approximately 360 mA of current (10 W) and can be powered by external batteries if desired. The major difference between the Tinman detector and the Tin-II detector is that Tin-II has larger volume ${}^3\text{He}$ counters to increase their efficiency. This change was made because of the anticipated lower count rate at Los Alamos altitudes compared to airplane flight altitudes. The detector uses two identical cylindrical ${}^3\text{He}$ ion chambers. The two model #252 detectors were purchased from LND, Inc. (Oceanside, NY). The specifications are given in Appendix A. ${}^3\text{He}$ detectors were chosen because they have excellent efficiency for thermal neutrons while being particularly insensitive to neutrons of higher energies and gamma rays.

Two identical detectors approximately 2.5 cm diameter and 24 cm long are used. One detector is shielded with cadmium and one is unshielded. Because Cd has a very large absorption cross section for thermal neutrons, it effectively blocks thermal neutrons from the detector. The difference in count rates between the two detectors is used to determine the contribution from thermal neutrons. The thickness of the Cd shielding is 0.05 cm. With the cadmium absorption cross section of 2520 b for thermal neutrons, we get an attenuation of approximately $2.8 \cdot 10^{-3}$ for this thickness of cadmium.

Figure 1 shows a high-level electronics drawing of the signal processing electronics. The ${}^3\text{He}$ detector is powered through a Cremat (West Newton, MA) CR-110 preamp and is operated at +1150 V as recommended by the manufacturer. The preamp is mounted on a Cremat CR-150-R5-CSP evaluation board which provides the power and input/output circuitry for the preamp. The output pulse from the preamp is approximately 15 mV high, has a rise time of $\sim 1 \mu\text{s}$ and a fall time of approximately 200 μs . The output of the preamp passes through a pulse Shaper/Amplifier (Cremat CR-200-8 μs -R2.1). The Shaper/Amplifier is mounted on a CR-160-R7 evaluation board which provides signal gain, DC level and pole-zero adjustments. The Shaper/Amplifier converts the output of the preamp to a Gaussian shaped pulse with a FWHM of approximately 15 μs and is approximately 5 volts high. An important feature of the shaper is that it stabilizes the baseline and lets the discriminator operate at the several 100 mV level. The gains of the two detectors were matched by looking at their pulse height spectra with a multichannel pulse-height analyzer. Following the Shaper/Amplifier, the pulse is input into the

discriminator circuit that produces a TTL logic pulse when the input pulse exceeds the voltage level of the discriminator. The discriminator level is adjustable between 0 and 5 volts.

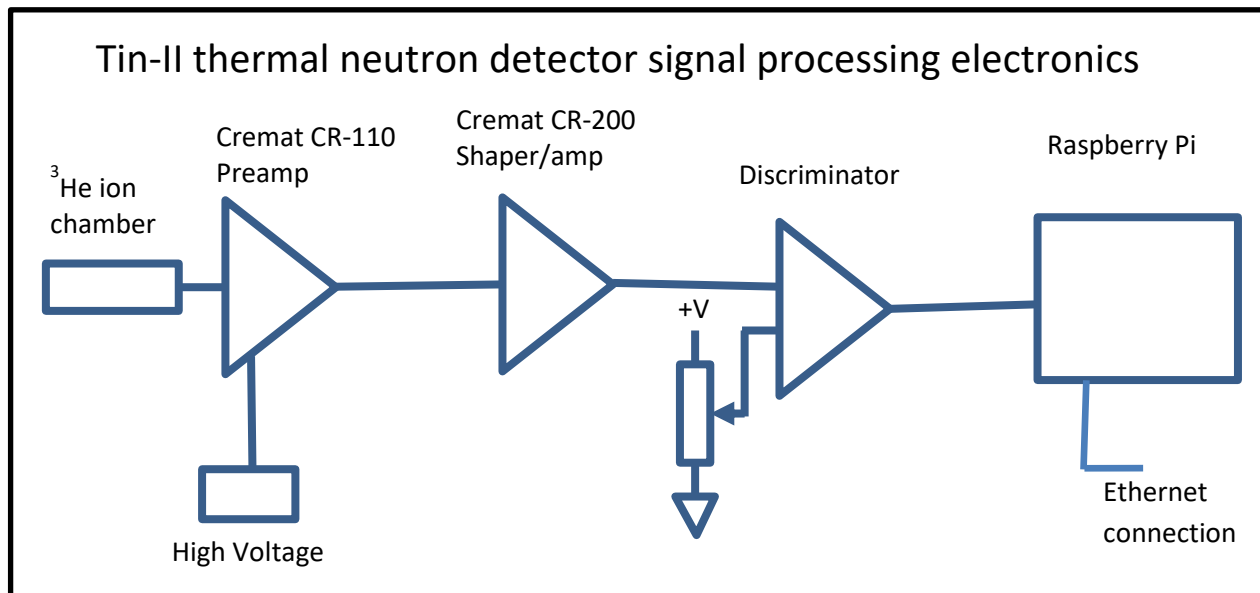


Figure 1. The signal processing electronics diagram of the Tin-II thermal-neutron detector

Figures 2 and 3 shows the pulse height spectrum of the two ^3He detectors from a moderated Pu-Be neutron source. The peak in the spectrum corresponds to the ejected proton and tritium ions which are produced following neutron reactions on ^3He depositing their full energy in the counter gas.

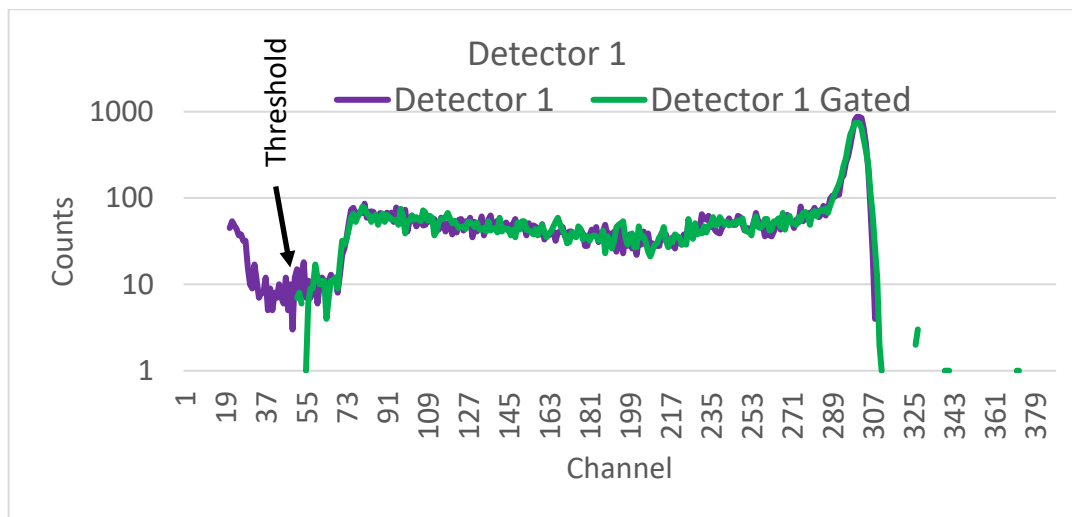


Figure 2. Pulse height spectrum of the ^3He Detector 1 using a moderated thermal-neutron source. The purple line is ungated and the green line is gated by the discriminator.

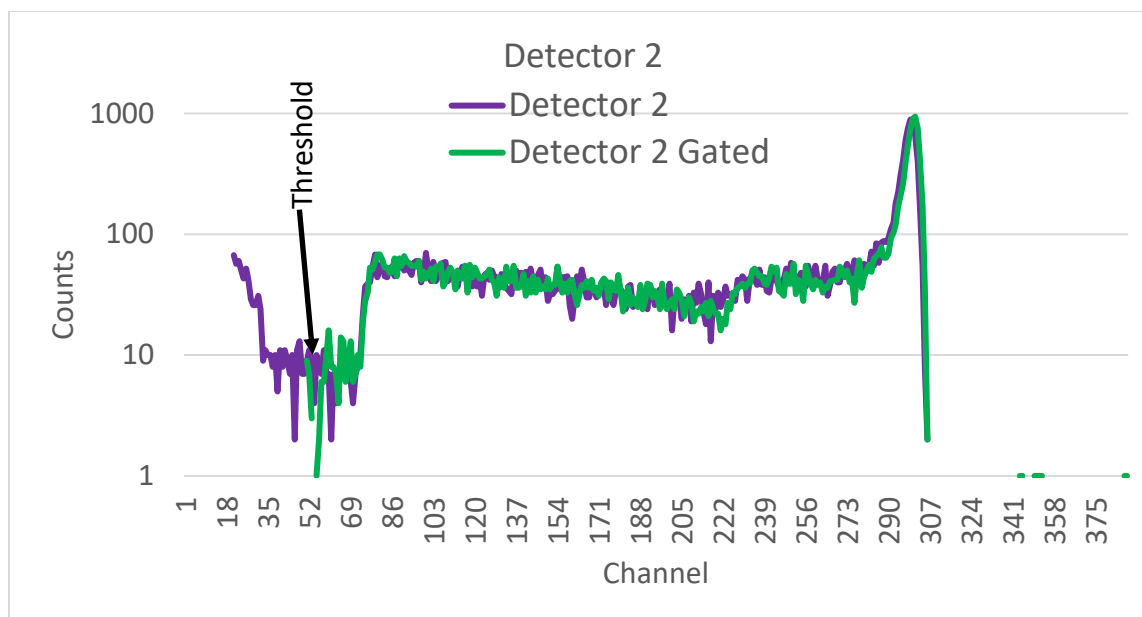


Figure 3. Pulse height spectrum of the ^3He Detector 2 using a moderated thermal-neutron source. The purple line is ungated and the green line is gated by the discriminator.

The counts to the left of the peak are events where the ejected ions hit the walls and do not deposit their full energy in the gas. Also shown in figures 2 and 3 are the pulse height spectra gated by the discriminator. We set the discriminator level to be in region above the noise but below the energy of the reaction products at approximately channel 50. The discriminator is set to 0.75 V.

Fabrication of thermal-neutron detector

The thermal-neutron detector was fabricated by LANL/ISR staff to meet the mechanical and electrical specifications of the NASA aircraft. The material of the box containing the detectors and the electronics is approximately 0.63 cm thick aluminum. This thickness of aluminum will attenuate thermal neutrons by approximately 5%.

Figure 4 shows the detector box with the internet cable on the left, the power cable on the right and the power switch and indicator lights. The detector box is 38 cm x 38 cm and 7.6 cm deep. Figure 5 shows the inside of the detector box with the lid rotated up.

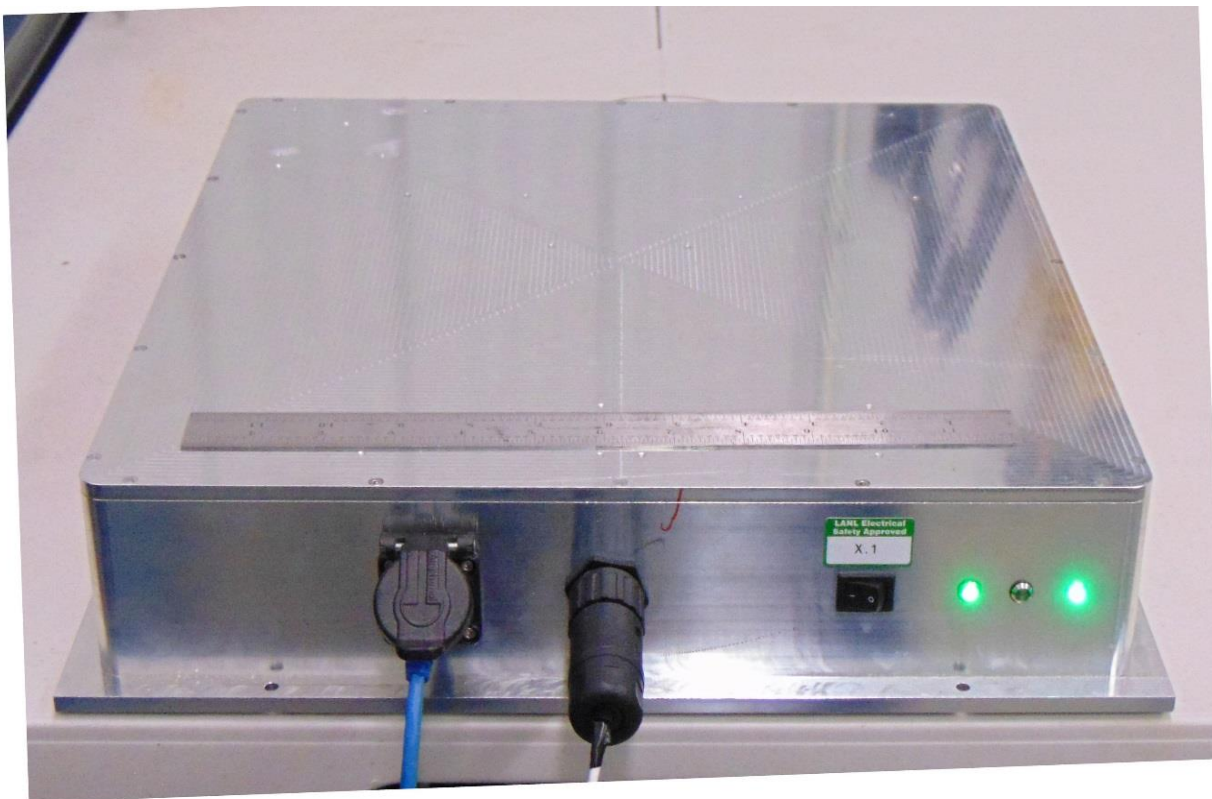


Figure 4. The Tin-II detector enclosure with the Ethernet and power cables connected. The scale on top of the detector is 30.5 cm long.

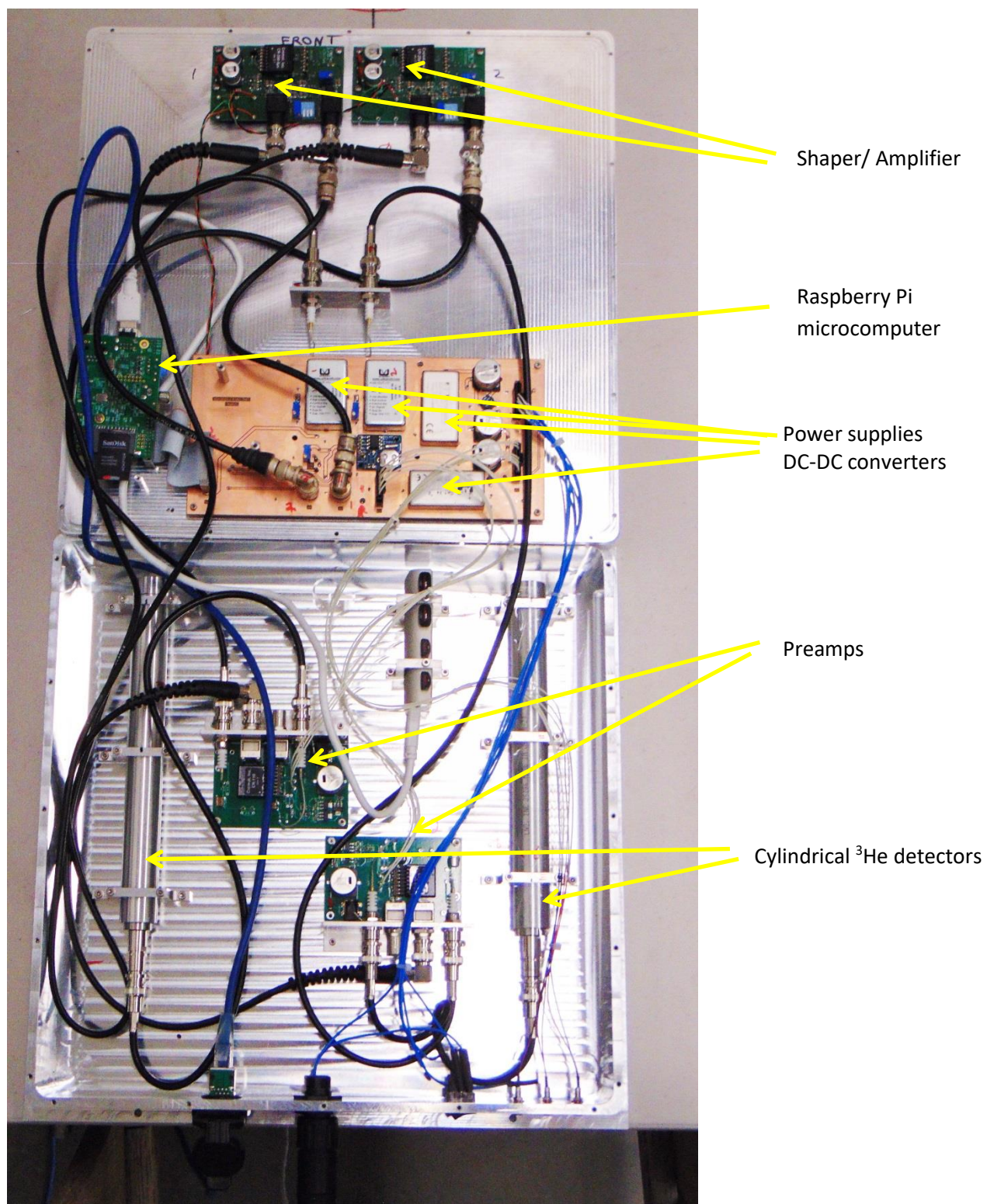


Figure 5. Inside of the Tin-II thermal-neutron detector. The Raspberry Pi microcomputer, the power supplies, the pre-amps and the discriminator circuits are attached to the lid of the box. The cylindrical ^3He detectors and the Shaper/Amplifiers are mounted to the bottom of the box. In this picture, the cadmium shield is not on the detector.

Data acquisition software

The data acquisition software runs on a Raspberry-Pi (R-Pi) microcomputer and starts automatically as a daemon on boot. The R-Pi has a LINUX based operating system (Raspbian). The WiringPI GPIO library (<http://wiringpi.com>) together with custom C++ code is used to detect the TTL logic pulses as interrupts on the GPIO pins. The full DAQ software can be found in Appendix 2. The timestamp of the event in unixtime is recorded as well as the number of the detector. A DT1307 real-time clock is used to maintain system time when Tin-II is not on the network. The internal R-Pi watchdog is enabled so that if the system becomes non-responsive for more than 15 seconds, it is rebooted. While the DAQ software is running, it maintains the watchdog. After a fixed time (nominally 10 minutes), the DAQ closes, the watchdog is no longer maintained, and the system automatically reboots, restarting the DAQ.

The TTL logic outputs from the two ^3He detector discriminators are input to the R-Pi computer. When the R-Pi receives this interrupt signal from the detector, it records a time stamp of the event to 100 μs precision and information about which detector produced the interrupt. Lists of time stamps and detector identifications are stored in files that are filled for 10 minutes. After 10 minutes, the files are closed and a new file is opened. These data are stored on 4 memory sticks for redundancy. In addition to the detectors, a heartbeat signal triggers the R-Pi every 5 seconds to determine that the program is operating correctly. The date and time are supplied to the R-Pi with a real-time clock. The time stamps can be sorted by detector and subsequently binned and presented as a histogram of count rate vs. date/time for each detector for any time bin width chosen.

The system was designed so that if there are any power interruptions, the system will restart automatically. Data files can be retrieved via SSH protocol communication with the system (WINSXP or equivalent can be used). The DAQ is configured on a private network with IP:127.0.1.1 and username:pi. If necessary, the memory sticks can be removed from the detector and the data downloaded without the R-Pi operating. It is assumed that four memory sticks will be formatted as fat32/vfat. The DAQ process is owned by root, so no special ownership or write privileges are required on the memory sticks.

Results

Our analysis of the data assumes that the relative acceptance of the two detectors are the same. We can measure the relative acceptance of the two detectors by removing the cadmium shielding from Detector 2 and counting ambient background. Measuring the relative acceptance of the two detectors is crucial to determining the thermal-neutron intensity. Figure 6 shows the count rate for the two detectors with the cadmium shield removed from detector 2 so both detectors are unshielded.

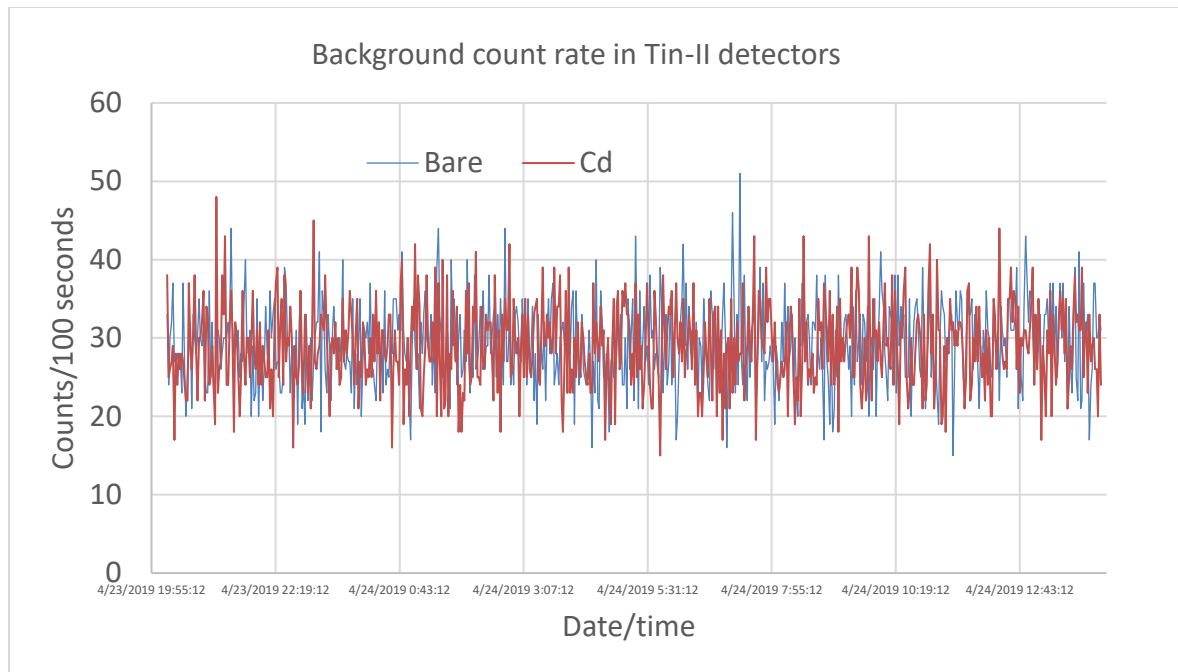


Figure 6. The count rate in 100 sec bins over time for the detector 0 (blue) and detector 1 (red) thermal-neutron detectors when both are unshielded.

The data were acquired over an 18-hour period. As seen in the plot, the count rates are essentially the same in both detectors with an average count rate of 29.2 ± 0.2 counts per 100 sec in Detector 1 and

28.9 ± 0.2 in Detector 2. The average difference is approximately 1% with Detector 1 having approximately 1% more counts than Detector 2. This difference can be attributed to variations in the detectors and thresholds. We can correct for this difference if necessary. This is roughly the count rate we should expect in the HPC area.

In figure 7, we show the count rate for the unshielded detector (blue line) and the cadmium covered detector (red line). As seen in the figure, the count rate in the cd-shielded detector is significantly less than the unshielded detector. The difference between the bare and the shielded detectors is the contribution from thermal neutrons.

The data shown in figure 7 were taken over a 20 hour run at TA-53 in building MPF-17. The average count rate for the bare detector was 30.2 ± 0.2 counts/100 sec and agrees with the previous measurement in Detector 1 within 3%. The count rate in the cadmium-shielded detector was 4.99 ± 0.08 counts/100 sec. The net number of counts/sec due to thermal neutrons is the difference between these rates or 25.2 ± 0.2 counts/ 100 sec. The non-thermal count rate (cadmium detector) is 17% of the total rate (in bare detector).

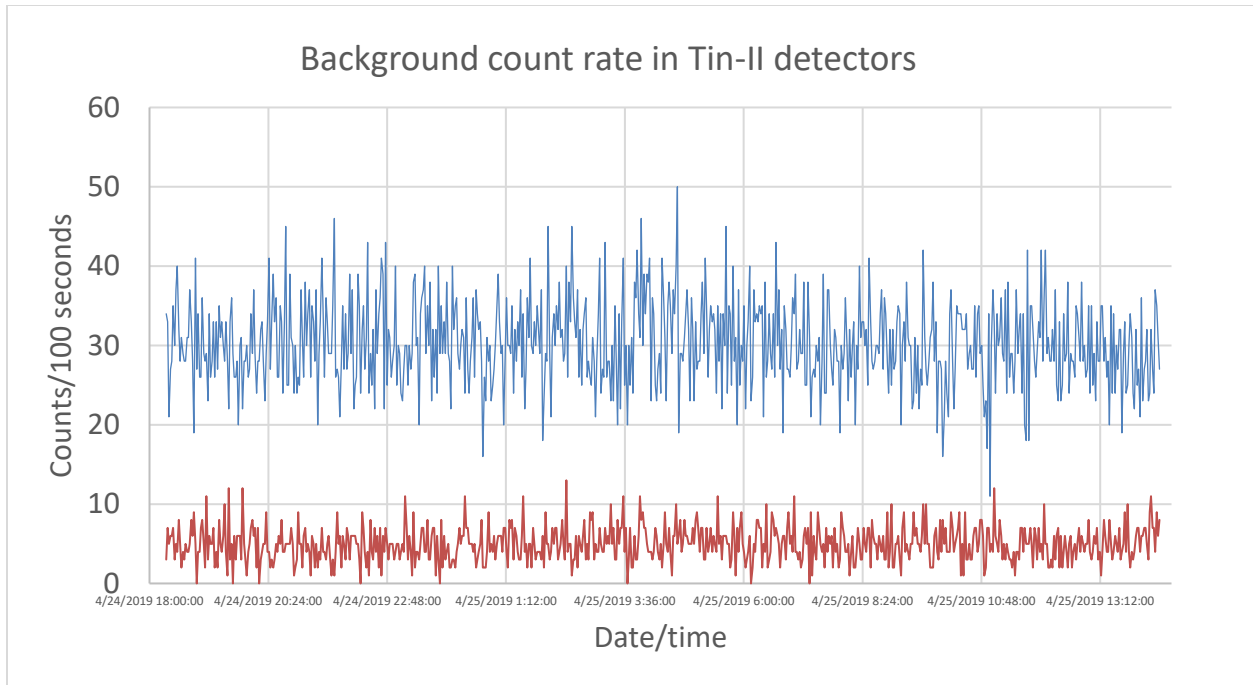


Figure 7. The count rate in time in 100 second bins in the bare detector (Blue line) and the Cd shielded detector (red line)

To convert counts in the detector to number of neutrons/cm² you need to know the detector acceptance. The efficiency of the detector can be estimated from the specification given by the manufacturer. From the specifications give in Appendix A, the sensitivity is 28 counts/second/nv where n is the density of neutrons in neutrons/cm³ and v is the velocity in cm/sec. As seen below, these units are the same as counts/sec/neutron/cm²/sec.

$$\text{cps/nv} = \frac{\text{counts}}{\text{sec} * \frac{\text{neutron}}{\text{cm}^3} * \frac{\text{cm}}{\text{sec}}} = \frac{\text{counts/ sec}}{\text{neutron /cm}^2/\text{sec}}$$

Since the sensitivity given by the manufacturer is 28 counts/sec/n/cm²/sec, we can convert the counts/sec in our detectors to number of n/cm²/ hr. Figure 8 shows the number of thermal neutrons/cm²/hr obtained by subtracting the counts/s in Detector 2 (Cd shielded) from Detector 1 (Bare) and including the sensitivity from the manufacturer and expressing the rate per hour.

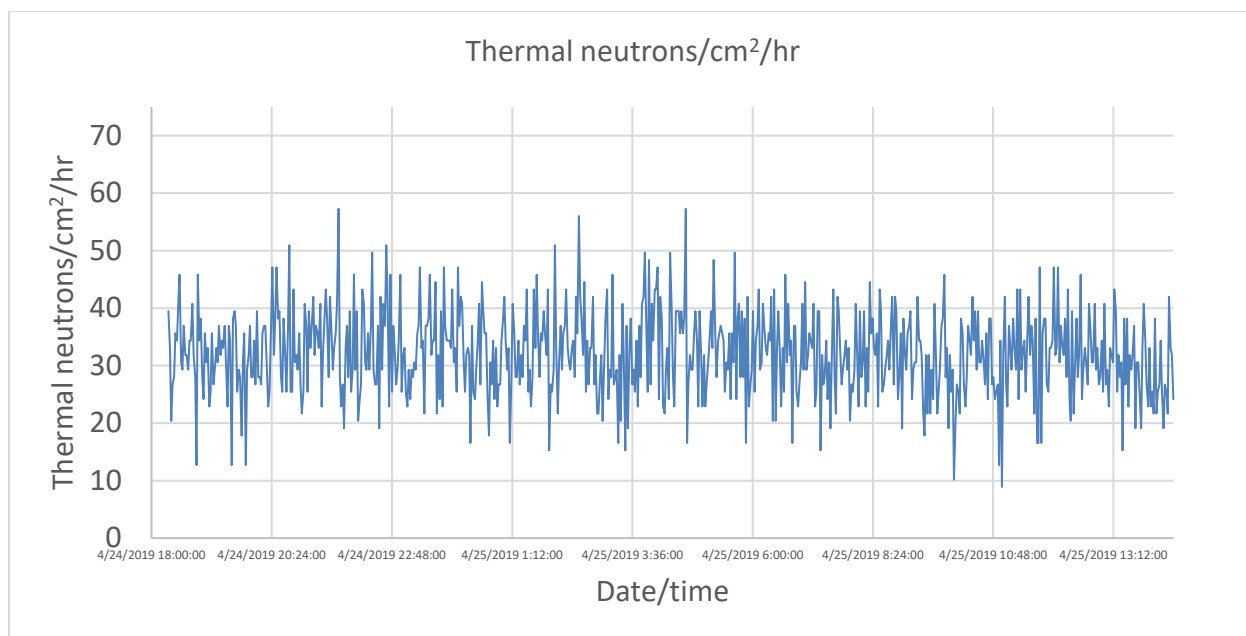


Figure 8 shows the number of thermal neutrons/cm²/hr as a function of time.

The average value of the number of thermal neutrons/cm²/hr that we measure with Tin-II is 32.1 +/- 0.3. This number seems reasonable given the literature value of the thermal neutron rate is ~5 thermal neutrons/cm²/hr at sea level. We must correct for the differences in the high-energy neutron flux at Los Alamos compared to sea level. The ratio of the high-energy neutron flux between Los Alamos and sea level is thought to be 5.6. This gives a corrected “literature” value of the thermal rate to be [5 thermal neutrons/cm²/hr]*5.6 which is approximately 28 thermal neutrons/cm²/hr. This number can be compared to the 32.1 thermal neutrons/cm²/hr that we measured. With these assumptions, our measurements give a value for the thermal neutron flux approximately 14% greater than the corrected “literature” value.

We operated Tin-II detector for several weeks at TA-53, MPF-17 and analyzed the results. Figure 9 shows the average thermal-neutron flux on the east bench, the west bench and the floor in TA-53, MPF-17. Clearly there is a different thermal-neutron environment on the west bench and the floor from the east bench. In a few runs the measured neutron intensity is well beyond statistical errors but within 15% of the average values. Improved analysis may reduce these fluctuations.

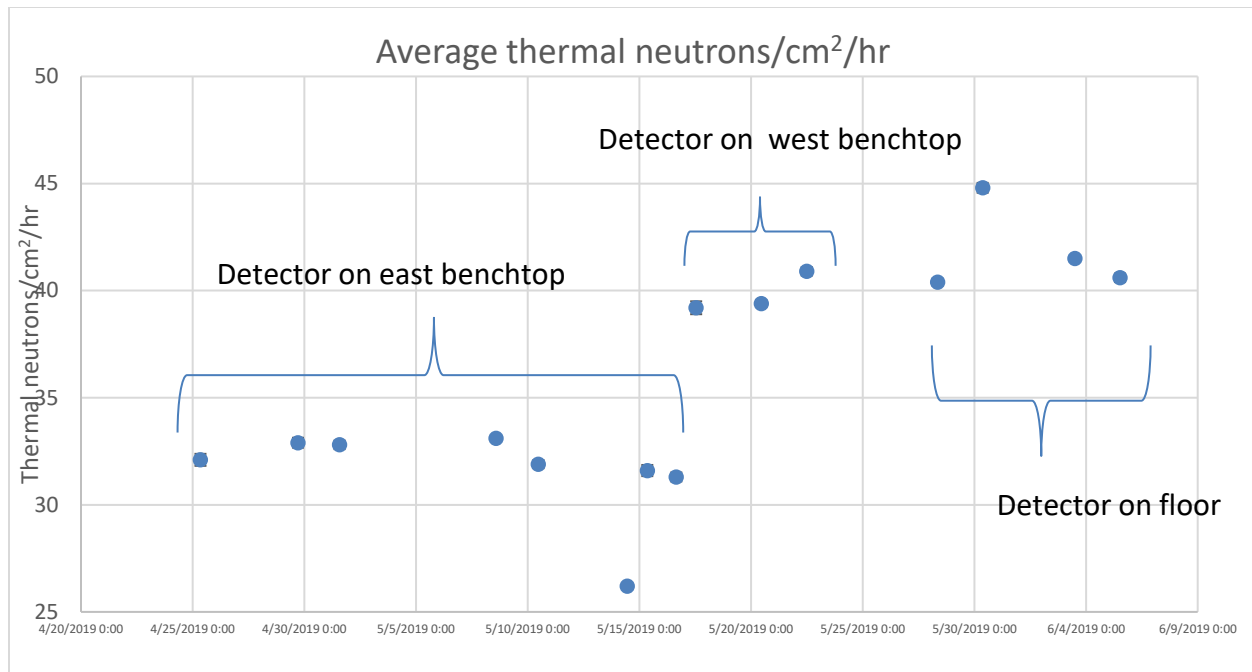


Figure 9 Average thermal-neutron fluxes over several week time period and different locations.

Occasionally we observed “spikes” in the data. The average count rate in a typical run the bare detector is approximately 0.3 counts/s or 1 count in 3.3 sec. In a spike, the count rate jumps to approximately 2500 cts/sec or approximately 400 μ s between pulses. These spikes are not considered neutron capture events and are either noise in the detector or other particles (muons?) causing this level of ionization in the counters. We suggest that these spurious events be removed from the data with software filters.

Proposed improvements

The Tinman detector was designed to operate in low count rate situations. Typical count rates are less than 1 count/sec. There have been recent interest in using this type of detector in higher count rate situations such as in the Lujan Center ER-2 experimental area to measure the ambient thermal neutron intensity. To operate at higher count rates the following changes should be considered:

1. Using shorter integration times in the Shaper/Amplifier. Tin-II presently has an 8 μ s shaping time. Shaping times of 50, 100, 250, 500, 1000, 2000, and 4000 ns can be obtained from Cremat. This requires using a different chip in the evaluation board.
2. Since the discriminator produces a positive output when the output of the Shaper/Amplifier is above the discriminator threshold voltage, the width of the TTL logic pulse from the discriminator circuit depends on the width of the linear input pulse. To provide a pulse wide enough to trigger the R-Pi, it may be necessary to retrigger these shorter pulses with a one-shot to provide a fixed width pulse.

3. At higher count rates it may be necessary to have active baseline restoration. Such a device is supplied by Cremat (CR-210) which is a module that plugs into the Shaper/Amplifier board.
4. One should consider whether the R-Pi provides sufficient capability to operate at higher count rates. It is possible that the computer should be upgraded to provide a faster and more robust operating capability.

Conclusion

The Tin-II detector should work well in the HPC computer area to monitor the relative intensity of thermal neutrons. We have estimated the absolute neutron intensity based on the given manufacturer's sensitivity. The efficiency of the detector should be measured with a calibrated thermal neutron source that is available at TA-36. This measurement would increase the confidence in the operation of this detector.

The analysis process can be improved with software filtering of the data to remove data spikes.

To predict the upset rate due to thermal neutrons it is necessary to know the upset cross section for particular devices to thermal neutrons. This cross section for HPC devices can and should be measured at the low-energy neutron source at LANSCE at the Lujan Center.

Appendix A



☎ 516-678-6141
✉ info@LNDinc.com



Search

CONTACT US

[Home](#) [About](#) [Product Line ▼](#) [Resources ▼](#) [Contact](#)

[LND | Nuclear Radiation Detectors](#) > [Products](#) > [Neutron Detectors](#) > [He3 Detectors](#) > [Cylindrical He3 Neutron Detectors](#) > 252

252

Cylindrical he3 neutron detector

☐ Compare [Request a quote](#)

[Download Specifications in .pdf format](#)

GENERAL SPECIFICATIONS

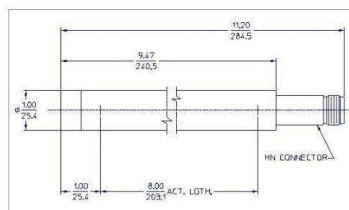
MAXIMUM LENGTH (INCH/MM)	11.20/284.5
MAXIMUM DIAMETER (INCH/MM)	1.0/25.4
EFFECTIVE LENGTH (INCH/MM)	8.0/203.2
CATHODE MATERIAL (INTERNAL/EXTERNAL)	Aluminum
EFFECTIVE DIAMETER (INCH/MM)	0.93/23.62
CONNECTOR	HN
OPERATING TEMPERATURE RANGE °C	-50 to +100
GAS PRESSURE (TORR)	3040
EFFECTIVE VOLUME (CM ³)	89.01

ELECTRICAL SPECIFICATIONS

RECOMMENDED OPERATING VOLTAGE (VOLTS)	1150
OPERATING VOLTAGE RANGE (VOLTS)	1050 - 1400
MAXIMUM PLATEAU SLOPE (%/100 VOLTS)	1
TUBE CAPACITANCE (PF)	8
WEIGHT (GRAMS)	142
MAXIMUM RESOLUTION (% FWHM)	6

THERMAL NEUTRON SENSITIVITY

SENSITIVITY (CPS/NV)	28.0
----------------------	------



Appendix B

```
//
// devel_daq.cxx
// author: a. couture
// description: This is a code designed to read the interrupts for a
//              Rpi and record the time. It presumes two detectors.
//              It is build of off primitive daq, but is presumes
//              there are not limit switches. It does, however,
//              expect 4 data storage locations.
//
//

//These are for daemon land
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <syslog.h>

// These are for watchdog
#include <linux/watchdog.h>
#include <sys/ioctl.h>

//These are for everything else
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <errno.h>
#include <fstream>
#include <iomanip>
#include <iostream>
#include <sstream>
#include <string>
#include <sys/time.h>
#include <wiringPi.h>

using namespace std;

// 3He detector 0:
// For now, use GPIO 17 for Interrupt, Pin 0 for Wiring Pi, Header pin 11
#define DET0_PIN 0 // header position 11

// 3He detector 1--Cd:
// For now, use GPIO 27 for Interrupt, Pin 2 for Wiring Pi, Header pin 13
#define DET1_PIN 2 // header position 13
```

```

// Create an event counter
volatile int detCounter[2] = { 0, 0 };

namespace{
    const int Nfiles = 4;
    ofstream datafile[ Nfiles ];
    ofstream monitorfile, logfile;
    const int status_time = 5; // (sec) how often to report time to data file
    // const int restart_time = 20; // (min) how often to restart runs
    const int restart_time = 10; // (min) how often to restart runs
}

//-----
// Define a function to be called when an
// interrupt is received for a detector or if an update is wanted
void write_event( int detector )
{
    static struct timeval tvNow;
    gettimeofday( &tvNow, 0 ); //get time in UTC
    for ( int ii = 0; ii < ::Nfiles; ++ii )
        if ( ::datafile[ii].is_open() )
            ::datafile[ii] << tvNow.tv_sec << "." << setw(6) << setfill('0') << tvNow.tv_usec
                << "    " << detector << endl;
    ++detCounter[ detector ];
}

//-----
// Define a function to be called when an
// interrupt is received for a detector or if an update is wanted
void det0_event( void )
{
    int detector = 0;
    write_event( detector );
}

//-----
// Define a function to be called when an
// interrupt is received for a detector or if an update is wanted
void det1_event( void )
{
    int detector = 1;
    write_event( detector );
}

// Here we have the detector monitoring
int devel_daq( int watchdogHandle )
{
    ::logfile << "Made it in to devel_daq" << endl;
}

```

```

// set up data logging
struct timeval runTime;
gettimeofday( &runTime , 0 );
for ( int ii = 0; ii < ::Nfiles; ++ii )
{
    ostringstream dataName;
    dataName << "data/" << ii << "/time_" << runTime.tv_sec << ".dat";
    ::datafile[ii].open( ( dataName.str() ).c_str() );
}
::logfile << " Successfully created data files" << endl;
// we write an initial event to get things started
write_event( -1 );
::logfile << " Initial write succeeded" << endl;

if ( wiringPiSetup() < 0 )
{
    ::logfile << "Unable to setup wiringPi: " << strerror( errno ) << endl;
    return EXIT_FAILURE;
}
::logfile << " WiringPi setup succeeded" << endl;

if ( wiringPiISR( DET0_PIN, INT_EDGE_RISING, &det0_event ) < 0 )
{
    ::logfile << "Unable to setup ISR: " << strerror( errno ) << endl;
    return EXIT_FAILURE;
}
::logfile << " det0 ISR setup succeeded" << endl;

if ( wiringPiISR( DET1_PIN, INT_EDGE_RISING, &det1_event ) < 0 )
{
    ::logfile << "Unable to setup ISR: " << strerror( errno ) << endl;;
    return EXIT_FAILURE;
}
::logfile << " det1 ISR setup succeeded" << endl;

// display counter once per second
int loop = 0;
::monitorfile.open( "/tmp/present_count.log" );
while ( loop < ( 60 * ::restart_time / status_time ) )
{
    // feed the watchdog--note problems if status_time > actual_timeout
    ioctl( watchdogHandle, WDIOC_KEEPALIVE, 0);
    ::monitorfile << detCounter[ 0 ] << " " << detCounter[ 1 ] << endl ;
    delay( status_time * 1000 ); // write down the time as a system check
    write_event( -1 );
}

```

```

    ++loop;
}

for ( int ii = 0; ii < ::Nfiles ; ++ii )
    ::datafile[ii].close();
::monitorfile.close();
return EXIT_SUCCESS;
::logfile << "Leaving devel_daq" << endl;

}

int init_watchdog()
{
    int watchdogHandle;
    if ( ( watchdogHandle = open("/dev/watchdog", O_RDWR | O_NOCTTY ) ) < 0 )
    {
        printf("Error: Couldn't open watchdog device! %d\n", watchdogHandle);
        return -1;
    }

    int desired_timeout = 15;
    int actual_timeout;
    if ( desired_timeout > 16 )
    {
        // desired_timeout is greater than hardware max. Resetting.
        // Should this be an assert instead???
        desired_timeout = 16;
    }
    ioctl(watchdogHandle, WDIOC_SETTIMEOUT, &desired_timeout);
    ioctl(watchdogHandle, WDIOC_GETTIMEOUT, &actual_timeout);

    if ( actual_timeout != desired_timeout )
    {
        // Unable to properly set timeout
        return -2;
    }

    return watchdogHandle;
}

int main()
{
    // Not quite sure where to start the watchdog--
    // may not get heartbeats from here, but we'll try

    ::logfile.open( "/tmp/daq_d.log" );
    int watchdogHandle;
    watchdogHandle = init_watchdog();

```

```

if ( watchdogHandle < 0 )
{
    ::logfile << "Problem opening watchdog with return code"
        << watchdogHandle << endl;
    exit ( EXIT_FAILURE );
}

pid_t pid, sid;

pid = fork();
if ( pid < 0 )
{
    // fork failed--get out of here
    exit( EXIT_FAILURE );
}

if ( pid > 0 )
{
    // fork succeeded--kill the parent
    exit( EXIT_SUCCESS );
}

// not sure on this one...
umask(022);

// Should there be logging here?

// create sid for the child
sid = setsid();
if ( sid < 0 )
{
    //failed--no sid
    exit( EXIT_FAILURE );
}

//change the working dir to an existant place
if ( ( chdir( "/" ) ) < 0 )
{
    //couldn't cd...
    exit( EXIT_FAILURE );
}

close( STDIN_FILENO );
close( STDOUT_FILENO );
close( STDERR_FILENO );

//do something

```

```
::logfile << "Got this far" << endl;

while ( devel_daq( watchdogHandle ) == EXIT_SUCCESS )
{
    continue;
}

::logfile.close();

// Note: we close the watchdog without disabling it
// as we should never get here--this will (hoepfully)
// cause a reboot

close( watchdogHandle );

exit( EXIT_FAILURE );

}
```